委　任　状

２００１年５月３０日

私儀　弁理士渡辺秀治、同長谷川洋、同青木修を代理人と定めて下記の権限を委任します。

１．特許協力条約に基づく国際出願
「システム開発支援装置、システム開発支援方法、およびコンピュータ読み取り可能な記録媒体」に関する一切の件

２．上記出願及び指定国の指定を取下げる件

３．上記出願についての国際予備審査の請求に関する一切の件並びに請求及び選択国の選択を取下げる件

あて名　東京都世田谷区北沢３－５－１８
名　称　株式会社　鷹山
代表者　高取　直

あて名　東京都世田谷区北沢３－５－１８　株式会社鷹山内
氏　名　半間　謙太郎

THIS PAGE BLANK (USPTO)

明 細 書

システム開発支援装置、システム開発支援方法、およびコンピュータ読み取り可能な記録媒体

技術分野

本発明は、ハードウェア部分とソフトウェア部分とが混在するシステムを開発する際に使用されるシステム開発支援装置、システム開発支援方法およびコンピュータ読み取り可能な記録媒体に関するものである。

背景技術

例えば携帯電話機といった電子機器を開発する際には、その機能をハードウェアとソフトウェアの両方を利用して実現することが多い。

第７図は、ハードウェア部分とソフトウェア部分とが混在する電子機器内のシステム１０１の一例を示すブロック図である。

第７図に示すシステム１０１において、ＭＰＵ（Micro Processing Unit）１１１は、システム１０１のソフトウェア部分としてＲＯＭ１１３、ＲＡＭ１１４およびフラッシュＲＯＭ１１５に記憶されたプログラムを実行する演算装置である。

また、ＤＳＰ（Digital Signal Processor）１１２は、特定の処理をハードウェアとして実現した回路である。

さらに、ＲＯＭ１１３は、プログラムやデータを予め記憶しているメモリである。ＲＡＭ１１４は、プログラム実行時に、そのプログラムやデータなどを一次的に記憶するメモリである。フラッシュＲＯＭ１１５

は、不揮発性のメモリであって、製品出荷後にその内容を書き換え可能なメモリである。

さらに、レジスタ群１１６は、プログラム実行時に各種データを保持する回路である。

さらに、ゲートアレイ１１７は、システム１０１のハードウェア部分として実装される論理回路である。

さらに、周辺回路１１８は、例えば図示せぬ周辺装置の制御や他の装置とのデータの授受などを行う回路である。

このようなシステム１０１を有する電子機器では、周辺回路１１８により得られた情報やユーザの指令に応じた処理を、ＭＰＵ１１１がプログラムに従って実行したり、ゲートアレイ１１７が実行したり、あるいは、両者が連携して実行したりする。

次に、このようなシステム１０１を開発する際の、従来のシステム開発方法について説明する。第８図は、従来のシステム開発方法を説明するフローチャートである。

まず、従来のシステム開発方法では、電子機器のシステム１０１に所望される機能、その機能のうちのハードウェアとして実現する部分およびソフトウェアとして実現する部分の指定、並びに、使用するＣＰＵコア、ゲートアレイの種類などを含む基本仕様が、文章や図面として策定される（ステップＳ１０１）。このような基本仕様のうち、ハードウェアとして実現する部分およびソフトウェアとして実現する部分の指定、並びに、使用するＣＰＵコア、ゲートアレイの種類などを決定するには様々な知識が必要であるため、それらの決定は、高度な知識を有する熟練者により行われることが多い。

次に、基本仕様のうちのソフトウェア部分に対応する論理仕様が、ソ

フトウェア開発者により、Ｃ言語などの高級言語でプログラムとして記述される（ステップＳ１１１）。そして、このプログラムがコンパイルされて、オブジェクトモジュールが生成される（ステップＳ１１２）。さらに、このオブジェクトモジュールに、必要に応じてライブラリ内のモジュールをリンクして、実行形式モジュールを生成する（ステップＳ１１３）。

一方、基本仕様のうちのハードウェア部分に対応する論理仕様が、ハードウェア開発者により、ＨＤＬ（Hardware Description Language）などの言語でプログラムとして記述される（ステップＳ１２１）。そして、このプログラムがコンパイルされて（ステップＳ１２２）、ＲＴＬ（Register Transfer Level）などの言語で回路仕様を記述したプログラムが生成され、この回路仕様を記述したプログラムから回路レイアウトが生成される（ステップＳ１２３）。

このように、ソフトウェア部分の論理仕様からシステム１０１のソフトウェア部分が生成され、ハードウェア部分の論理仕様からシステム１０１のハードウェア部分が生成される。

そして、予め基本仕様から生成された検証プログラムを使用して、このシステム１０１のソフトウェア部分とハードウェア部分の検証が実行される（ステップＳ１１４、ステップＳ１２４）。

ソフトウェア部分とハードウェア部分の検証結果のそれぞれが良好であるか否か、すなわち、仕様どおりに動作しているか否かが判断される（ステップＳ１１５、ステップＳ１２５）。

検証結果が良好ではない場合、その検証結果に応じて、ソフトウェアの論理設計の段階（ステップＳ１１１）やハードウェアの論理設計の段階（ステップＳ１２１）に戻り、それぞれ、論理設計を修正したり、場合によっては基本設計を変更したりする。

そして、ソフトウェア部分およびハードウェア部分ともに良好な検証結果が得られるまで、ソフトウェア開発者とハードウェア開発者とが、試行錯誤を繰り返して、論理設計あるいは基本設計を修正していく。

この際、ソフトウェア開発者は、ソフトウェアに比べハードウェアについての知識をあまり持っておらず、反対に、ハードウェア開発者は、ハードウェアに比べソフトウェアについての知識をあまり持っていないために、ソフトウェア開発者とハードウェア開発者との協調作業が難しく、この修正には、一般的に多くに時間が必要とされる。

そして、最終的に良好な検証結果が得られた後、得られたシステムの論理に基づいて、システム１０１がＩＣチップとして生成される（ステップＳ１０２）。

しかしながら、従来のシステム開発方法では、上述のように、ソフトウェア開発者とハードウェア開発者との協調作業が難しく、この設計の修正に多くの時間が必要とされるため、システムの完成までの時間が長くなってしまうという問題がある。

本発明は、上記の問題を解決するためになされたものであり、システムの完成までの時間の短縮を可能にするためのシステム開発支援装置、システム開発支援方法およびコンピュータ読み取り可能な記録媒体を得ることを目的とする。

発明の開示

本発明のシステム開発支援装置は、プログラムの各部をハードウェア部分およびソフトウェア部分のいずれかに指定する切り分け情報に基づ

いて、システムの論理仕様が単一の高級言語で記述されているプログラムをハードウェア部分とソフトウェア部分とに切り分ける切り分け手段と、その切り分け手段により切り分けられたハードウェア部分のプログラムおよびソフトウェア部分のプログラムを記憶する記憶手段と、その記憶手段に記憶されたハードウェア部分のプログラムを回路仕様に変換する第１の変換手段と、その記憶手段に記憶されたソフトウェア部分のプログラムを実行形式モジュールに変換する第２の変換手段とを備えている。

このシステム開発支援装置を利用すると、ハードウェア部分とソフトウェア部分とが混在するシステムの完成までの時間を短縮することができる。

さらに、本発明のシステム開発支援装置は、上記発明のシステム開発支援装置に加え、切り分け手段が、切り分け情報に基づいて、単一の高級言語で記述されているプログラムの機能ブロックごとに、ハードウェアとして実装する部分であるか、ソフトウェアとして実装する部分であるかを決定するようにしたものである。

このシステム開発支援装置を利用すると、さらに、ハードウェアとして実装する部分と、ソフトウェアとして実装する部分とを適切に切り分けることができる。

さらに、本発明のシステム開発支援装置は、上記各発明のシステム開発支援装置に加え、システムの仕様に基づいて切り分け情報を生成する切り分け情報生成手段を備えている。

このシステム開発支援装置を利用すると、さらに、システム開発者が熟練者でなくても適切な切り分け情報を生成することができる。

さらに、本発明のシステム開発支援装置は、上記各発明のシステム開発支援装置に加え、切り分け情報生成手段が、システムにおいて実行形式モジュールが記憶されるメモリの容量、およびシステムにおいて回路仕様に基づく回路が実行されるゲートアレイのゲート数に基づいて、あるいは、メモリの容量およびゲート数とともに、システムにおいて使用されるＣＰＵコアの種類、システムにおいて使用されるＤＳＰの機能、使用可能なハードウェアマクロおよび使用可能なソフトウェアマクロの少なくとも１つに基づいて、切り分け情報を生成するようにしたものである。

このシステム開発支援装置を利用すると、さらに、システムの仕様におけるこれらのパラメータに基づいて切り分け情報を生成することで、より適切な切り分け情報を生成することができる。

さらに、本発明のシステム開発支援装置は、上記各発明のシステム開発支援装置に加え、第１の変換手段により変換された回路仕様に基づく回路、および第２の変換手段により変換された実行形式モジュールの動作を検証する検証手段を備えている。

このシステム開発支援装置を利用すると、さらに、論理仕様を記述された目標プログラムから生成されたシステムの論理全体が一括して検証され、ハードウェア部分およびソフトウェア部分のそれぞれの動作検証に加え、両者の協調に基づく動作についての検証をも行うことができる。

さらに、本発明のシステム開発支援装置は、上記各発明のシステム開発支援装置に加え、検証手段による検証結果に応じて、切り分け情報を変更する切り分け情報変更手段を備えている。

このシステム開発支援装置を利用すると、さらに、システム開発者による切り分け情報の設定／変更の頻度が減少し、特に希少な熟練者の作業量を低減することができるとともに、システムの開発に要する時間をより短くすることができる。

さらに、本発明のシステム開発支援装置は、上記各発明のシステム開発支援装置に加え、切り分け情報変更手段が、検証手段による検証結果に応じて、ハードウェア部分とソフトウェア部分との比率を変更するようにしたものである。

このシステム開発支援装置を利用すると、さらに、システムのハードウェア条件（メモリ容量やゲート数など）を特に変更せずに、そのハードウェア条件に合致した回路が設計される。

さらに、本発明のシステム開発支援装置は、上記各発明のシステム開発支援装置に加え、検証手段による検証結果に応じて、システムのハードウェア条件を変更する第１の条件変更手段を備え、第１の変換手段が、システムのハードウェア条件に応じて、ハードウェア部分のプログラムを回路仕様に変換するようにしたものである。

このシステム開発支援装置を利用すると、さらに、システム開発者によるハードウェア条件の変更頻度が減少し、特に希少な熟練者の作業量を低減することができるとともに、システムの開発に要する時間をより短くすることができる。

さらに、本発明のシステム開発支援装置は、上記各発明のシステム開発支援装置に加え、第１の条件変更手段が、検証手段による検証結果に応じて、ハードウェア部分とソフトウェア部分との間の信号の入出力タ

イミングを変更するようにしたものである。

このシステム開発支援装置を利用すると、さらに、ハードウェア部分とソフトウェア部分との間における信号伝達に起因する動作不良を回避することができる。

さらに、本発明のシステム開発支援装置は、上記各発明のシステム開発支援装置に加え、検証手段による検証結果に応じて、第２の変換手段がソフトウェア部分のプログラムを実行形式モジュールに変換する際のコンパイル条件を変更する第２の条件変更手段を備えている。

このシステム開発支援装置を利用すると、さらに、システム開発者によるソフトウェアのコンパイル条件の変更頻度が減少し、特に希少な熟練者の作業量を低減することができるとともに、システムの開発に要する時間をより短くすることができる。

さらに、本発明のシステム開発支援装置は、上記各発明のシステム開発支援装置に加え、第２の条件変更手段が、検証手段による検証結果に応じて、システムにおいて使用されるＣＰＵコアの種類を変更するようにしたものである。

このシステム開発支援装置を利用すると、さらに、ソフトウェア部分全体の動作速度を調節することができる。

さらに、本発明のシステム開発支援装置は、上記各発明のシステム開発支援装置に加え、所定の検証結果が得られるまで、あるいは、所定の反復回数だけ、切り分け情報、第１の変換手段がハードウェア部分のプログラムを回路仕様に変換する際のハードウェアの条件、および第２の変換手段がソフトウェア部分のプログラムを実行形式モジュールに変換

する際のコンパイル条件のうちの少なくとも１つを変更しつつ、切り分け手段、第１の変換手段、第２の変換手段および検証手段を繰り返し動作させる最適化手段を備えている。

このシステム開発支援装置を利用すると、さらに、システム開発者による切り分け情報の設定／変更の頻度が減少し、特に希少な熟練者の作業量を低減することができるとともに、システムの開発に要する時間をより短くすることができる。

本発明のシステム開発支援方法は、プログラムの各部をハードウェア部分およびソフトウェア部分のいずれかに指定する切り分け情報に基づいて、システムの論理仕様が単一の高級言語で記述されているプログラムをハードウェア部分とソフトウェア部分とに切り分け、そのハードウェア部分のプログラムおよびそのソフトウェア部分のプログラムを記憶手段に記憶させるステップと、その記憶手段に記憶された上記ハードウェア部分のプログラムを回路仕様に変換するステップと、その記憶手段に記憶された上記ソフトウェア部分のプログラムを実行形式モジュールに変換するステップとを備えている。

このシステム開発支援方法を利用すると、ハードウェア部分とソフトウェア部分とが混在するシステムの完成までの時間を短縮することができる。

本発明のコンピュータ読み取り可能な記録媒体に記録されたシステム開発支援プログラムは、プログラムの各部をハードウェア部分およびソフトウェア部分のいずれかに指定する切り分け情報に基づいて、システムの論理仕様が単一の高級言語で記述されているプログラムをハードウェア部分とソフトウェア部分とに切り分け、そのハードウェア部分のプ

ログラムおよびそのソフトウェア部分のプログラムを記憶手段に記憶させる切り分け手段、記憶手段に記憶された上記ハードウェア部分のプログラムを回路仕様に変換する第１の変換手段、並びに、記憶手段に記憶された上記ソフトウェア部分のプログラムを実行形式モジュールに変換する第２の変換手段としてコンピュータを機能させる。

このシステム開発支援プログラムを利用すると、ハードウェア部分とソフトウェア部分とが混在するシステムの完成までの時間を短縮することができる。

本発明のコンピュータ読み取り可能な記録媒体に記録された切り分けプログラムは、プログラムの各部をハードウェア部分およびソフトウェア部分のいずれかに指定する切り分け情報に基づいて、システムの論理仕様が単一の高級言語で記述されているプログラムをハードウェア部分とソフトウェア部分とに切り分け、そのハードウェア部分のプログラムおよびそのソフトウェア部分のプログラムを記憶手段に記憶させる切り分け手段としてコンピュータを機能させる。

この切り分けプログラムを利用すると、ハードウェア部分とソフトウェア部分とが混在するシステムの論理仕様を単一の高級言語で記述することが可能になり、システムの開発効率が向上する。

図面の簡単な説明

第１図は、本発明の実施の形態１に係るシステム開発支援装置の構成を示すブロック図である。

第２図は、第１図に示すシステム開発支援装置の動作について説明するフローチャートである。

第３図は、実施の形態１のシステム開発支援装置を使用した場合のシステムの開発の手順について説明するフローチャートである。

第４図は、本発明の実施の形態２に係るシステム開発支援装置の構成を示すブロック図である。

第５図は、本発明の実施の形態３に係るシステム開発支援装置の構成を示すブロック図である。

第６図は、第５図に示すシステム開発支援装置の動作について説明するフローチャートである。

第７図は、ハードウェア部分とソフトウェア部分とが混在する電子機器内のシステムの一例を示すブロック図である。

第８図は、従来のシステム開発方法を説明するフローチャートである。

発明を実施するための最良の形態

以下、本発明の実施の形態を図面に基づいて具体的に説明する。

実施の形態１.

第１図は、本発明の実施の形態１に係るシステム開発支援装置の構成を示すブロック図である。

第１図において、コンピュータ１は、システム開発支援プログラム２１を実行してシステム開発支援装置として機能する装置である。また、ディスプレイ２は、コンピュータ１の描画回路１６からの信号に応じた画像を表示する装置である。さらに、入力装置３は、キーボード、マウスなどといった開発者により操作され、その操作に応じた信号をコンピュータ１に供給する装置である。

コンピュータ１において、ＣＰＵ１１は、図示せぬオペレーティングシステム、システム開発支援プログラム２１などのプログラムを実行す

るものである。また、ＲＯＭ１２は、コンピュータ１の起動に必要なデータやプログラムなどを予め記憶したメモリであり、ＲＡＭ１３は、システム開発支援プログラム２１などのプログラムの実行中に、そのプログラムやデータを一時的に記憶する記憶手段としてのメモリである。

さらに、ハードディスクドライブ（以下、ＨＤＤという）１４は、システム開発支援プログラム２１や、その他、図示せぬオペレーティングシステムなどを格納する記録媒体を有する装置である。なお、これらのプログラムを格納しておく記録媒体は、磁気記録媒体であるＨＤＤに限定されるものではなく、可搬性のあるフレキシブルディスク、コンパクトディスクといった磁気ディスク、光ディスク、光磁気ディスクなどでもよい。

ＨＤＤ１４に格納されたシステム開発支援プログラム２１は、切り分けプログラム３１、コンパイラプログラム３２、コンパイラプログラム３３、リンカプログラム３４および検証プログラム３５を含むプログラムである。

この切り分けプログラム３１は、プログラムの各部をハードウェア部分およびソフトウェア部分のいずれかに指定する切り分け情報に基づいて、システムの論理仕様が単一の高級言語で記述されているプログラムをハードウェア部分とソフトウェア部分とに切り分け、そのハードウェア部分のプログラムおよびそのソフトウェア部分のプログラムをＲＡＭ１３またはＨＤＤ１４に記憶させる切り分け手段としてコンピュータ１を機能させるプログラムである。

このコンパイラプログラム３２は、ＲＡＭ１３またはＨＤＤ１４に記憶されたハードウェア部分の高級言語のプログラムを回路仕様に変換する第１の変換手段としてコンピュータ１を機能させるプログラムである。

このコンパイラプログラム３３およびリンカプログラム３４は、ＲＡ

ＭＲＡＭ１３またはＨＤＤ１４に記憶されたソフトウェア部分の高級言語のプログラムを実行形式モジュールに変換する第２の変換手段としてコンピュータ１を機能させるプログラムである。

なお、このうちのコンパイラプログラム３３は、ＲＡＭ１３またはＨＤＤ１４に記憶されたソフトウェア部分のプログラムをオブジェクトモジュールに変換するためのプログラムであり、リンカプログラム３４は、そのオブジェクトモジュールから実行形式モジュールを生成したり、あるいはそのオブジェクトモジュールおよび図示せぬライブラリや他のオブジェクトモジュールをリンクして、実行形式モジュールを生成したりするためのプログラムである。

この検証プログラム３５は、システムの論理仕様に対応する検証仕様に基づいて生成され、コンパイラプログラム３２により変換された回路仕様に基づく回路およびコンパイラプログラム３３およびリンカプログラム３４により変換された実行形式モジュールの動作を検証する検証手段としてコンピュータ１を機能させるためのプログラムである。

さらに、インターフェース１５は、ＨＤＤ１４との間でデータの授受を行う回路である。

さらに、描画回路１６は、供給されるデータに応じてディスプレイ２に画像信号を供給して画像を表示させる回路である。

さらに、インターフェース１７は、入力装置３からの信号を取得する回路である。

さらに、インターフェース１８は、図示せぬ外部装置との間でデータの授受を行う回路である。

次に、このシステム開発支援装置としてのコンピュータ１の動作について説明する。第２図は、第１図に示すシステム開発支援装置の動作に

ついて説明するフローチャートである。

まず、システム開発者により例えばＣ言語といった単一の高級言語で作成された、システムを記述したプログラムが、例えばＨＤＤ１４やＲＡＭ１３に用意される。

また、システム開発者により、プログラムの各部をハードウェア部分およびソフトウェア部分のいずれかに指定する切り分け情報が例えばＨＤＤ１４やＲＡＭ１３に用意される（ステップＳ１）。

なお、切り分け情報には、例えば、所定の機能ブロック（システムにおいて所定の機能を実現する１または複数のルーチンで構成される群）ごとに、その部分をハードウェアとして実現するか、あるいはソフトウェアとして実現するかを指定する情報が含まれる。また、ハードウェアおよびソフトウェアのいずれでもよい場合には、その旨を指定するか、あるいは特に何も指定しないようにする。なお、この切り分け情報をＨＤＤ１４などには記憶させずに、システム開発支援プログラム２１の実行時に、パラメータとして与えるようにしてもよい。

そして、システム開発者の操作に応じて、またはシステム開発支援プログラム２１の実行時に自動的に、ＣＰＵ１１は、システム開発支援プログラム２１のうちの切り分けプログラム３１を実行する。

ＣＰＵ１１は、切り分けプログラム３１に従って、単一の高級言語でシステムが記述されたプログラム（以下、目標プログラムという）を読み込み（ステップＳ２）、切り分け情報を参照して、その目標プログラムの各部分を、ハードウェア部分およびソフトウェア部分のいずれかに振り分ける（ステップＳ３）。

この際、例えば、その目標プログラムにおける各機能ブロックが、ハードウェア部分およびソフトウェア部分のいずれかに振り分けられる。機能ブロックごとに目標プログラムを切り分ける場合、例えば、処理速

度が要求される機能を実現する機能ブロックなどがハードウェア部分に割り振られる。

例えば、目標プログラムを作成する前、すなわち基本仕様において、実現する各機能とその機能ブロックの名前との関係を決めておき、その機能ブロックの名前のルーチン内で、その機能ブロックで実現する機能に関するプログラムを記述して目標プログラムを作成する。そして、切り分け情報には、その機能ブロックごとに、その機能ブロックの名前とハードウェア部分かソフトウェア部分かを指定する情報（以下、指定情報という）との組が設定される。これにより、ＣＰＵ１１は、切り分けプログラム３１に従って、切り分け情報に設定されている機能ブロックの名前と同一の名前のルーチンを発見すると、切り分け情報中のその機能ブロックの名前についての指定情報に基づいて、その機能ブロックのルーチンをハードウェア部分かソフトウェア部分かに振り分ける。

なお、ここでは、一例として、機能ブロック単位で目標プログラムを切り分ける例を示したが、目標プログラムを他の単位で切り分けてもよいし、切り分ける方法についても他の方法でもよい。

ＣＰＵ１１は、切り分けプログラム３１に従って、このように切り分けたハードウェア部分のプログラム（すなわち、ハードウェアとして実現される１または複数のルーチン）のファイル、およびソフトウェア部分のプログラム（すなわち、ソフトウェアとして実現される１または複数のルーチン）のファイルを、ＲＡＭ１３に記憶させたり、ＨＤＤ１４に格納させたりする。

次に、ＣＰＵ１１は、コンパイラプログラム３２を実行する。ＣＰＵ１１は、コンパイラプログラム３２に従って、ハードウェア部分の高級言語のプログラムを、ＲＴＬなどの回路仕様に対応した言語のプログラムにコンパイルする（ステップＳ４）。この回路仕様に対応した言語のプ

ログラムは、一旦、ＲＡＭ１３またはＨＤＤ１４に記憶される。

このハードウェア部分のコンパイルの際には、使用されるゲートアレイの種類、ゲート数の上限、ＩＣチップ作成の際に使用するプロセスの種類、ＩＣチップ量産時のテスト回路の種類（このテスト回路の種類によりＩＣチップのピン配置が制限される）などのハードウェア条件がコンパイル条件として参照される。このハードウェア条件は、コンパイラプログラム３２実行時にシステム開発者が入力するようにしてもよいし、予め、ファイルなどに記述しておくようにしてもよい。

また、この際、必要に応じて、各プログラム部分間の信号の授受の関係の情報などの制約条件をファイルなどに別途記述しておくようにしてもよい。その場合、ＣＰＵ１１は、コンパイラプログラム３２に従って、ハードウェア部分とソフトウェア部分との境界における信号の授受の関係などの制約条件を満足する回路仕様のプログラムを生成する。

さらに、ＣＰＵ１１は、コンパイラプログラム３３を実行する。ＣＰＵ１１は、コンパイラプログラム３３に従って、ソフトウェア部分の高級言語のプログラムをオブジェクトモジュールにコンパイルする（ステップＳ５）。

このソフトウェア部分のコンパイルの際には、使用されるＣＰＵコアの種類、最適化オプションなどのコンパイル条件が参照される。このコンパイル条件は、コンパイラプログラム３３実行時にシステム開発者が入力するようにしてもよいし、予め、ファイルなどに記述しておくようにしてもよい。

この際、必要に応じて、各プログラム部分間の信号の授受の関係の情報などの制約条件をファイルなどに別途記述しておくようにしてもよい。その場合、ＣＰＵ１１は、コンパイラプログラム３３に従って、ハードウェア部分とソフトウェア部分との境界における信号の授受の関係など

の制約条件を満足するように、例えばソフトウェア部分のプログラムを適宜修正して、オブジェクトモジュールを生成する。

その後、ＣＰＵ１１は、リンカプログラム３４を実行する。ＣＰＵ１１は、リンカプログラム３４に従って、目標プログラムのソフトウェア部分のオブジェクトモジュールと、図示せぬライブラリに登録されているモジュール、その他のオブジェクトモジュールをリンクして、実行形式モジュールを生成する（ステップＳ６）。

なお、ここでは、ハードウェア部分のプログラムをコンパイルした後に、ソフトウェア部分のプログラムをコンパイルし、リンクしているが、先に、ソフトウェア部分のプログラムをコンパイルし、リンクした後に、ハードウェア部分のプログラムをコンパイルするようにしてもよい。また、ハードウェア部分のプログラムのコンパイルと、ソフトウェア部分のプログラムのコンパイルおよびリンクとを並行して実行するようにしてもよい。

このようにして、ハードウェア部分とソフトウェア部分とが混在したシステム全体の論理が生成される。この論理の検証のために、ハードウェア部分の回路仕様が具体化される。この際の具体化としては、例えば、ハードウェア部分の回路仕様に基づいてその回路をシミュレートするシミュレータプログラムや、論理の再構築が可能なゲートアレイなどで試作した回路が挙げられる。シミュレータプログラムを用いる場合には、ＣＰＵ１１がそのシミュレータプログラムに従って、ＲＡＭ１３やＨＤＤ１４に記憶された回路仕様に基づいて、回路のシミュレーションを実行する。また、試作回路を用いる場合には、その回路をインターフェース１８を介して接続する。

そして、ＣＰＵ１１は、検証プログラム３５を実行する。ＣＰＵ１１は、検証プログラム３５に従って、ハードウェア部分とソフトウェア部

分とが混在して生成されたシステム全体の論理に対して、各種入力を行い、そのときの各部位における論理、すなわち信号の挙動、出力、結果などを取得し、その入力と挙動、出力、結果などとの関係が所定の条件を満たしているか否かを判断して、システム全体の論理を検証する（ステップＳ７）。

なお、検証プログラム３５に従って、ＣＰＵ１１が、この検証の結果をディスプレイ２に表示したり、図示せぬプリンタに印刷させたりするようにしてもよい。

次に、実施の形態１のシステム開発支援装置を使用した場合のシステムの開発の手順について説明する。第３図は、実施の形態１のシステム開発支援装置を使用した場合のシステムの開発の手順について説明するフローチャートである。

実施の形態１のシステム開発支援装置を使用してシステム開発を行う場合、第３図に示すように、まず、システム開発者により基本仕様が設計される（ステップＳ２１）。この基本仕様としては、各種機能の仕様のみが決定される。すなわち、この基本仕様では、ハードウェアとして実現する部分およびソフトウェアとして実現する部分の指定、並びに、使用するＣＰＵコア、ゲートアレイの種類などは、原則として決定されない。ただし、デフォルトとして、これらについて典型的なものを仮に定めておくようにしてもよい。

次に、システム開発者により、論理仕様が、その基本仕様から、単一の高級言語で記述された目標プログラムとして設計される（ステップＳ２２）。

また、システム開発者による手操作で、あるいは、システムが搭載される電子機器のハードウェア（メモリ容量、ゲート数など）の条件など

に基づいて自動的に、その目標プログラムに対する、最初の切り分け情報が設定される（ステップＳ２３）。

そして、システム開発支援プログラム２１が実行されると、コンピュータ１が上述のように動作し、システム全体の論理が生成され、その論理についての検証結果が得られる（ステップＳ２４）。

次に、この検証結果に基づいて、システム開発者は、検証結果が良好か否かを判断する（ステップＳ２５）。そして、その検証結果が良好である場合には、設計されたそのシステムを具体化したＩＣチップの作成が行われる（ステップＳ２６）。

一方、その検証結果が良好ではない場合には、システム開発者により、切り分け情報が変更されたり、論理仕様が変更されたり、場合によっては基本仕様が変更されたりする。そして、良好な検証結果が得られるまで、上述と同様に処理を繰り返す。

以上のように、上記実施の形態１によれば、システム開発支援プログラム２１に従って、コンピュータ１が、切り分け情報に基づいて、システムの論理仕様が単一の高級言語で記述されている目標プログラムをハードウェア部分とソフトウェア部分とに切り分け、そのハードウェア部分のプログラムを回路仕様に変換するとともに、そのソフトウェア部分のプログラムを実行形式モジュールに変換する。これにより、ハードウェア部分とソフトウェア部分とが混在するシステムの完成までの時間を短縮することができる。

すなわち、設計変更の際には、単一の言語で記述された目標プログラム、および切り分け情報を修正すればよいため、ハードウェア開発者とソフトウェア開発者との協調がほとんど必要なくなり、開発効率が向上する。

また、高度な知識を有する熟練者が基本仕様に関わる割合を減らすことが可能であり、開発者の人材活用の効率も向上する。

さらに、上記実施の形態１によれば、切り分けプログラム３１に従って、コンピュータ１が、切り分け情報に基づいて、目標プログラムの機能ブロックごとに、ハードウェアとして実装する部分であるか、ソフトウェアとして実装する部分であるかを決定するようにしたので、ハードウェアとして実装する部分と、ソフトウェアとして実装する部分とを適切に切り分けることができる。すなわち、機能ブロックごとに切り分けることで、例えば動作速度を要求される機能のルーチン群が一括してハードウェアとして実装され、逆に、ソフトウェアとして実現したほうが好ましい機能のルーチン群が一括してソフトウェアとして実装される。

さらに、上記実施の形態１によれば、検証プログラム３５に従って、コンピュータ１が、ハードウェア部分に対応する回路仕様に基づく回路、およびソフトウェア部分に対応する実行形式モジュールの動作を検証する。これにより、論理仕様を記述した目標プログラムから生成されたシステムの論理全体が一括して検証され、ハードウェア部分およびソフトウェア部分のそれぞれの動作検証に加え、両者の協調に基づく動作についての検証をも行うことができる。

実施の形態２.

本発明の実施の形態２に係るシステム開発支援装置は、実施の形態１に係るシステム開発支援装置のシステム開発支援プログラム２１に、システムの仕様に基づいて切り分け情報を生成する切り分け情報生成プログラム３６を追加したものである。

第４図は、本発明の実施の形態２に係るシステム開発支援装置の構成を示すブロック図である。第４図において、システム開発支援プログラム２１Ａは、実施の形態１のシステム開発支援プログラム２１に、システムの仕様に基づいて切り分け情報を生成する切り分け情報生成プログラム３６を追加したものである。

この切り分け情報生成プログラム３６は、システムの仕様に基づいて切り分け情報を生成する切り分け情報生成手段としてコンピュータ１Ａを機能させるためのプログラムである。

なお、第４図におけるその他の構成要素については、実施の形態１におけるものと同様であるので、その説明を省略する。

次に、上記装置の動作について説明する。

この切り分け情報生成プログラム３６は、最初の切り分け情報を生成する際、あるいは、切り分け情報を変更する際に、ＣＰＵ１１により実行される。

その場合、ＣＰＵ１１は、切り分け情報生成プログラム３６に従って、予め定められているシステムの仕様に基づいて切り分け情報を生成する。

例えば、ＣＰＵ１１は、切り分け情報生成プログラム３６に従って、システムが実現されるＩＣのチップサイズ、システムにおいて実行形式モジュールが記憶されるメモリ（ＲＯＭ１１３やフラッシュＲＯＭ１１５）の容量、システムにおいて回路仕様に基づく回路が実行されるゲートアレイ１１７のゲート数などのシステムの仕様に基づいて切り分け情報を生成する。

あるいは、例えば、ＣＰＵ１１は、切り分け情報生成プログラム３６に従って、そのチップサイズ、メモリの容量およびゲート数、並びに、システムにおいて使用されるＣＰＵコアの種類、システムにおいて使用

されるＤＳＰの機能、使用可能なハードウェアマクロおよび使用可能なソフトウェアマクロの少なくとも１つに基づいて、切り分け情報を生成する。

すなわち、切り分け情報生成プログラム３６には、上述のシステムの仕様のパラメータ（チップサイズなど）の値と、目標プログラムの所定の部分あるいは機能ブロックについての実現方式（ハードウェアまたはソフトウェア）との関係の知識が予め内蔵され、その知識に基づいて、システムの仕様から切り分け情報が生成される。

なお、その他の動作については実施の形態１と同様であるので、その説明を省略する。また、この実施の形態２では、実施の形態１に切り分け情報生成プログラム３６を追加して切り分け情報を自動生成するようにしたが、他の実施の形態に切り分け情報生成プログラム３６を追加して切り分け情報を自動生成することも勿論可能である。

以上のように、上記実施の形態２によれば、切り分け情報生成プログラム３６に従って、コンピュータ１Ａが、システムの仕様に基づいて切り分け情報を生成する。これにより、システム開発者が熟練者でなくても適切な切り分け情報を生成することができる。

また、上記実施の形態２によれば、切り分け情報生成プログラム３６に従って、コンピュータ１Ａが、システムにおいて実行形式モジュールが記憶されるメモリの容量、およびシステムにおいて回路仕様に基づく回路が実行されるゲートアレイのゲート数に基づいて、あるいは、メモリの容量およびゲート数とともに、システムにおいて使用されるＣＰＵコアの種類、システムにおいて使用されるＤＳＰの機能、使用可能なハードウェアマクロおよび使用可能なソフトウェアマクロの少なくとも１つに基づいて、切り分け情報を生成する。これにより、システムの仕様

におけるこれらのパラメータに基づいて切り分け情報を生成することで、より適切な切り分け情報を生成することができる。

実施の形態３．

本発明の実施の形態３に係るシステム開発支援装置は、実施の形態１に係るシステム開発支援装置のシステム開発支援プログラム２１に、検証プログラム３５による検証結果に応じて、切り分け情報などを変更して検証結果を最適化する最適化プログラム５１を追加したものである。

第５図は、本発明の実施の形態３に係るシステム開発支援装置の構成を示すブロック図である。第５図において、システム開発支援プログラム２１Ｂは、実施の形態１のシステム開発支援プログラム２１に、検証プログラム３５による検証結果に応じて、切り分け情報などを変更して検証結果を最適化する最適化プログラム５１を追加したものである。

なお、この最適化プログラム５１は、検証プログラム３５による検証結果に応じて、切り分け情報を変更する切り分け情報変更手段としてコンピュータ１Ｂを機能させるためのプログラムである。

また、この最適化プログラム５１は、検証プログラム３５による検証結果に応じて、ハードウェア部分とソフトウェア部分との比率を変更する切り分け情報変更手段としてコンピュータ１Ｂを機能させるためのプログラムである。

さらに、この最適化プログラム５１は、検証プログラム３５による検証結果に応じて、システムのハードウェア条件を変更する第１の条件変更手段としてコンピュータ１Ｂを機能させるためのプログラムである。

さらに、この最適化プログラム５１は、検証プログラム３５による検証結果に応じて、コンパイラプログラム３３によりソフトウェア部分のプログラムを実行形式モジュールに変換する際のコンパイル条件を変更

する第２の条件変更手段としてコンピュータ１Ｂを機能させるためのプログラムである。

さらに、この最適化プログラム５１は、所定の検証結果が得られるまで、あるいは、所定の反復回数だけ、切り分け情報、ハードウェア部分のプログラムを回路仕様に変換する際のハードウェア条件、およびソフトウェア部分のプログラムを実行形式モジュールに変換する際のコンパイル条件のうちの少なくとも１つを変更しつつ、切り分けプログラム３１、コンパイラプログラム３２、コンパイラプログラム３３、リンカプログラム３４および検証プログラム３５を繰り返し実行させる最適化手段としてコンピュータ１Ｂを機能させるためのプログラムである。

なお、第５図におけるその他の構成要素については、実施の形態１におけるものと同様であるので、その説明を省略する。

次に、上記装置の動作について説明する。第６図は、第５図に示すシステム開発支援装置の動作について説明するフローチャートである。

このシステム開発支援装置は、システム開発支援プログラム２１Ｂに従って、実施の形態１と同様にして、システム全体の論理を生成し、その論理の検証を行う（ステップＳ１～Ｓ７）。

次に、ＣＰＵ１１は、最適化プログラム５１を実行する。

まず、ＣＰＵ１１は、最適化プログラム５１に従って、検証結果が所定の条件を満足するか否かを判断する（ステップＳ４１）。

この際、例えば、各種信号の遅延時間が所定の値以内であるか、動作検証した機能の結果が所望のものになったかなどが判断される。

検証結果が所定の条件を満足すると判断した場合、ＣＰＵ１１は、最適化プログラム５１に従って、システムの回路仕様の設計を終了する。

一方、検証結果が所定の条件を満足しないと判断した場合、ＣＰＵ１

１は、最適化プログラム５１に従って、切り分け情報、ハードウェアのコンパイル条件、ソフトウェアのコンパイル条件などが所定の回数だけ変更されたか否かを判断する（ステップＳ４２）。

このとき、所定の回数だけ、切り分け情報、ハードウェアのコンパイル条件、ソフトウェアのコンパイル条件などが変更されたと判断した場合にもＣＰＵ１１は、最適化プログラム５１に従って、システムの回路仕様の自動設計を終了する。

一方、所定の回数だけ、まだ切り分け情報、ハードウェアのコンパイル条件、ソフトウェアのコンパイル条件などが変更されていないと判断した場合には、ＣＰＵ１１は、最適化プログラム５１に従って、切り分け情報、ハードウェアのコンパイル条件、およびソフトウェアのコンパイル条件のうちの少なくとも１つを変更する(ステップＳ４３～Ｓ４８)。

まず、ＣＰＵ１１は、最適化プログラム５１に従って、検証結果に基づいて、切り分け情報を変更するか否かを判断し（ステップＳ４３)、必要に応じて切り分け情報を変更する（ステップＳ４４)。

例えば、検証の結果、あるソフトウェア部分のルーチンまたは機能ブロックの処理速度が遅いためにシステムが動作不良となった場合には、ゲートアレイのゲート数に余裕があれば、そのルーチンまたは機能ブロックはハードウェア部分に変更される。また、その場合に、ゲートアレイのゲート数に余裕がないときには、ハードウェア部分のルーチンまたは機能ブロックのうちソフトウェア部分へ変更可能なものがソフトウェア部分へ変更されるとともに、その動作不良の原因になっているソフトウェア部分のルーチンまたは機能ブロックがハードウェア部分に変更される。なお、ゲートアレイの残余ゲート数は、実装可能ゲート数から、コンパイル後の現時点でのハードウェア部分に対するゲート数を減算すれば得られる。

また、例えば、コンパイル後の現時点でのハードウェア部分に対するゲート数が実装可能ゲート数を超えている場合には、いずれかのハードウェア部分のルーチンまたは機能ブロックがソフトウェア部分に変更される。この際、ソフトウェア部分に変更するルーチンまたは機能ブロックの選択は、システム開発者が行ってもよいし、そのルーチンまたは機能ブロックに対するゲート数などに応じて自動的に最適化プログラム５１に従って自動的に行われるようにしてもよい。

このように、この場合、ＣＰＵ１１は、最適化プログラム５１に従って、検証結果に応じて、ハードウェア部分とソフトウェア部分との比率を変更する。

次に、ＣＰＵ１１は、最適化プログラム５１に従って、検証結果に基づいて、ハードウェア部分のコンパイル条件、すなわちハードウェア条件を変更するか否かを判断し（ステップＳ４５）、必要に応じてハードウェア部分のコンパイル条件を変更する（ステップＳ４６）。

例えば、ハードウェア部分からソフトウェア部分への信号の入出力タイミングが合っていないことにより、ソフトウェア部分に動作不良が発生している場合には、信号を遅延させたりして、ハードウェア部分からソフトウェア部分への信号の入出力タイミングが変更される。

次に、ＣＰＵ１１は、最適化プログラム５１に従って、検証結果に基づいて、ソフトウェア部分のコンパイル条件を変更するか否かを判断し（ステップＳ４７）、必要に応じてソフトウェア部分のコンパイル条件を変更する（ステップＳ４８）。

例えば、検証結果に応じて、システムにおいて使用されるＣＰＵコアの種類が変更される。ソフトウェア部分のうちの大部分のルーチンまたは機能ブロックの動作が遅い場合には、ＣＰＵコアが処理速度の高いものに変更される。この際、実装可能なＣＰＵコアの種類および性能を予

め列挙しておく。そして、それらの中から、最適化プログラム５１に従ってＣＰＵコアが適宜選択される。

また、例えば、検証結果に応じて、コンパイル時の最適化オプションが変更される。最適化オプションとしては、サイズ重視のオプション、速度重視のオプションなどがある。

このように切り分け情報、ハードウェア条件、およびソフトウェアのコンパイル条件のうちの少なくとも１つが変更される。

なお、切り分け情報、ハードウェア条件、およびソフトウェアのコンパイル条件のいずれを変更するかは、反復回数などに応じて決定するようにしてもよい。すなわち、例えば、まず最初の所定回においては、ハードウェア条件およびソフトウェアのコンパイル条件のみを変更し、その後、切り分け情報のみを変更したりしてもよい。

このようにして、切り分け情報、ハードウェア条件、およびソフトウェアのコンパイル条件のうちの少なくとも１つが変更された後、再度、論理仕様を記述したプログラムに対して切り分け処理が行われ、切り分けられたハードウェア部分およびソフトウェア部分がそれぞれ処理され、システム全体の論理が生成される。

そして、ステップＳ４１で検証結果が所定の条件を満足すると判断されるか、あるいは、ステップＳ４２でこの反復処理の回数が所定の回数に達したと判断されるまで、この処理が反復して行われる。

次に、実施の形態３のシステム開発支援装置を使用した場合のシステムの開発の手順について説明する。実施の形態３のシステム開発支援装置を使用した場合のシステムの開発の手順は、実施の形態１の場合（第３図）と同様になる。ただし、実施の形態３では、最適化処理を行うことで、システム開発者による切り分け情報の設定／変更の頻度は減少す

る。

また、例えば、切り分け情報、コンパイル条件などの変更を、最初の所定の回数だけ、システム開発者による手作業で行い、その後、上述のような最適化プログラム５１による自動で行うようにしてもよい。

また、切り分け情報、コンパイル条件などの変更を、検出結果におけるエラーの数が所定の数以下に減少するまでシステム開発者による手作業で行い、その後、上述のような最適化プログラム５１による自動で行うようにしてもよい。

このようにして、効率的にシステムを開発することができる。

以上のように、上記実施の形態３によれば、最適化プログラム５１に従って、コンピュータ１Ｂは、検証プログラム３５による検証結果に応じて、所定の検証結果が得られるまで、あるいは、所定の反復回数だけ、切り分け情報、ハードウェアの条件、およびソフトウェアのコンパイル条件のうちの少なくとも１つを変更しつつ、目標プログラムに基づくシステムの論理全体の生成を反復する。これにより、システム開発者による切り分け情報の設定／変更の頻度が減少し、特に希少な熟練者の作業量を低減することができるとともに、システムの開発に要する時間をより短くすることができる。

さらに、上記実施の形態３によれば、最適化プログラム５１に従って、コンピュータ１Ｂは、検証プログラム３５による検証結果に応じて、切り分け情報を変更する。これにより、システム開発者による切り分け情報の設定／変更の頻度が減少し、特に希少な熟練者の作業量を低減することができるとともに、システムの開発に要する時間をより短くすることができる。

さらに、上記実施の形態３によれば、最適化プログラム５１に従って、コンピュータ１Ｂは、検証プログラム３５による検証結果に応じて、ハードウェア部分とソフトウェア部分との比率を変更する。これにより、システムのハードウェア条件（メモリ容量やゲート数など）を特に変更せずに、そのハードウェア条件に合致した回路が設計される。

さらに、上記実施の形態３によれば、最適化プログラム５１に従って、コンピュータ１Ｂは、検証プログラム３５による検証結果に応じて、ハードウェア部分のコンパイル時に参照されるハードウェア条件を変更する。これにより、システム開発者によるハードウェア条件の変更頻度が減少し、特に希少な熟練者の作業量を低減することができるとともに、システムの開発に要する時間をより短くすることができる。

また、この際、コンピュータ１Ｂは、検証結果に応じて、例えば、ハードウェア部分とソフトウェア部分との間の信号の入出力タイミングを変更する。これにより、ハードウェア部分とソフトウェア部分との間における信号伝達に起因する動作不良を回避することができる。

さらに、上記実施の形態３によれば、最適化プログラム５１に従って、コンピュータ１Ｂは、検証プログラム３５による検証結果に応じて、ソフトウェア部分のプログラムを実行形式モジュールに変換する際のコンパイル条件を変更する。これにより、システム開発者によるソフトウェアのコンパイル条件の変更頻度が減少し、特に希少な熟練者の作業量を低減することができるとともに、システムの開発に要する時間をより短くすることができる。

また、この際、コンピュータ１Ｂは、検証結果に応じて、システムにおいて使用されるＣＰＵコアの種類を変更する。これにより、ソフトウェア部分全体の動作速度を調節することができる。

なお、上記実施の形態１～３においては、目標プログラムに切り分け情報を明示的に記述せずに、切り分け情報を切り分けプログラム３１に供給するようにしているが、目標プログラムに切り分け情報を明示的に記述し、その切り分け情報に基づいて切り分けプログラム３１が、切り分け処理を行うようにしてもよい。

また、高級言語としては、Ｃ言語の他、Ｃ＋＋といったＣ言語から派生した言語や、Ｃ言語とは全く異なるプログラミング言語を使用するようにしても勿論よい。

さらに、上記実施の形態１～３では、一例として、開発されるシステムがＩＣチップとして実装される場合について述べたが、ＩＣチップを含む回路基板として実装されるシステムの設計にも適用することができる。

産業上の利用可能性

本発明によれば、ハードウェア部分とソフトウェア部分とが混在するシステムの完成までの時間の短縮を可能にするためのシステム開発支援装置、システム開発支援方法、およびコンピュータ読み取り可能な記録媒体を得ることができる。

請求の範囲

1．プログラムの各部をハードウェア部分およびソフトウェア部分のいずれかに指定する切り分け情報に基づいて、システムの論理仕様が単一の高級言語で記述されているプログラムをハードウェア部分とソフトウェア部分とに切り分ける切り分け手段と、

上記切り分け手段により切り分けられた上記ハードウェア部分のプログラムおよび上記ソフトウェア部分のプログラムを記憶する記憶手段と、

上記記憶手段に記憶された上記ハードウェア部分のプログラムを回路仕様に変換する第１の変換手段と、

上記記憶手段に記憶された上記ソフトウェア部分のプログラムを実行形式モジュールに変換する第２の変換手段と、

を備えることを特徴とするシステム開発支援装置。

2．プログラムの各部をハードウェア部分およびソフトウェア部分のいずれかに指定する切り分け情報に基づいて、システムの論理仕様が単一の高級言語で記述されているプログラムをハードウェア部分とソフトウェア部分とに切り分ける切り分け手段と、

上記切り分け手段により切り分けられた上記ハードウェア部分のプログラムおよび上記ソフトウェア部分のプログラムを記憶する記憶手段と、

上記記憶手段に記憶された上記ハードウェア部分のプログラムを回路仕様に変換する第１の変換手段と、

上記記憶手段に記憶された上記ソフトウェア部分のプログラムを実行形式モジュールに変換する第２の変換手段と、

を備え、

上記切り分け手段は、上記切り分け情報に基づいて、上記単一の高級言語で記述されているプログラムの機能ブロックごとに、ハードウェアとして実装する部分であるか、ソフトウェアとして実装する部分であるかを決定すること、

を特徴とするシステム開発支援装置。

3．前記システムの仕様に基づいて前記切り分け情報を生成する切り分け情報生成手段を備えることを特徴とする請求の範囲第１項記載のシステム開発支援装置。

4．前記システムにおいて前記実行形式モジュールが記憶されるメモリの容量、および前記システムにおいて前記回路仕様に基づく回路が実行されるゲートアレイのゲート数に基づいて、あるいは、上記メモリの容量および上記ゲート数とともに、前記システムにおいて使用されるＣＰＵコアの種類、前記システムにおいて使用されるＤＳＰの機能、使用可能なハードウェアマクロおよび使用可能なソフトウェアマクロの少なくとも１つに基づいて、前記切り分け情報を生成する切り分け情報生成手段を備えることを特徴とする請求の範囲第１項記載のシステム開発支援装置。

5．前記第１の変換手段により変換された回路仕様に基づく回路、および前記第２の変換手段により変換された実行形式モジュールの動作を検証する検証手段を備えることを特徴とする請求の範囲第１項記載のシステム開発支援装置。

6．前記検証手段による検証結果に応じて、前記切り分け情報を変更する切り分け情報変更手段を備えることを特徴とする請求の範囲第５項記載のシステム開発支援装置。

7．前記検証手段による検証結果に応じて、ハードウェア部分とソフトウェア部分との比率を変更する切り分け情報変更手段を備えることを特徴とする請求の範囲第５項記載のシステム開発支援装置。

8．前記検証手段による検証結果に応じて、前記第１の変換手段が前記ハードウェア部分を回路仕様に変換する際に参照するハードウェア条件を変更する第１の条件変更手段を備えることを特徴とする請求の範囲第５項記載のシステム開発支援装置。

9．前記検証手段による検証結果に応じて、前記第１の変換手段が前記ハードウェア部分を回路仕様に変換する際に参照するハードウェア条件を変更する第１の条件変更手段を備え、

上記第１の条件変更手段は、前記検証手段による検証結果に応じて、ハードウェア部分とソフトウェア部分との間の信号の入出力タイミングを変更すること、

を特徴とする請求の範囲第５項記載のシステム開発支援装置。

１０．前記検証手段による検証結果に応じて、前記第２の変換手段が前記ソフトウェア部分のプログラムを実行形式モジュールに変換する際のコンパイル条件を変更する第２の条件変更手段を備えることを特徴とする請求の範囲第５項記載のシステム開発支援装置。

１１．前記検証手段による検証結果に応じて、前記第２の変換手段が前記ソフトウェア部分のプログラムを実行形式モジュールに変換する際のコンパイル条件を変更する第２の条件変更手段を備え、

上記第２の条件変更手段は、前記検証手段による検証結果に応じて、前記システムにおいて使用されるＣＰＵコアの種類を変更すること、

を特徴とする請求の範囲第５項記載のシステム開発支援装置。

１２．所定の検証結果が得られるまで、あるいは、所定の反復回数だけ、前記切り分け情報、前記第１の変換手段が前記ハードウェア部分のプログラムを回路仕様に変換する際のハードウェアの条件、および前記第２の変換手段が前記ソフトウェア部分のプログラムを実行形式モジュールに変換する際のコンパイル条件のうちの少なくとも１つを変更しつつ、前記切り分け手段、前記第１の変換手段、前記第２の変換手段および前記検証手段を繰り返し動作させる最適化手段を備えることを特徴とする請求の範囲第５項記載のシステム開発支援装置。

１３．プログラムの各部をハードウェア部分およびソフトウェア部分のいずれかに指定する切り分け情報に基づいて、システムの論理仕様が単一の高級言語で記述されているプログラムをハードウェア部分とソフトウェア部分とに切り分けるステップと、

上記ハードウェア部分のプログラムを回路仕様に変換するステップと、

上記ソフトウェア部分のプログラムを実行形式モジュールに変換するステップと、

を備えることを特徴とするシステム開発支援方法。

１４．プログラムの各部をハードウェア部分およびソフトウェア部分の

いずれかに指定する切り分け情報に基づいて、システムの論理仕様が単一の高級言語で記述されているプログラムをハードウェア部分とソフトウェア部分とに切り分け、そのハードウェア部分のプログラムおよびそのソフトウェア部分のプログラムを記憶手段に記憶させる切り分け手段、

上記記憶手段に記憶された上記ハードウェア部分のプログラムを回路仕様に変換する第１の変換手段、並びに、

上記記憶手段に記憶された上記ソフトウェア部分のプログラムを実行形式モジュールに変換する第２の変換手段、

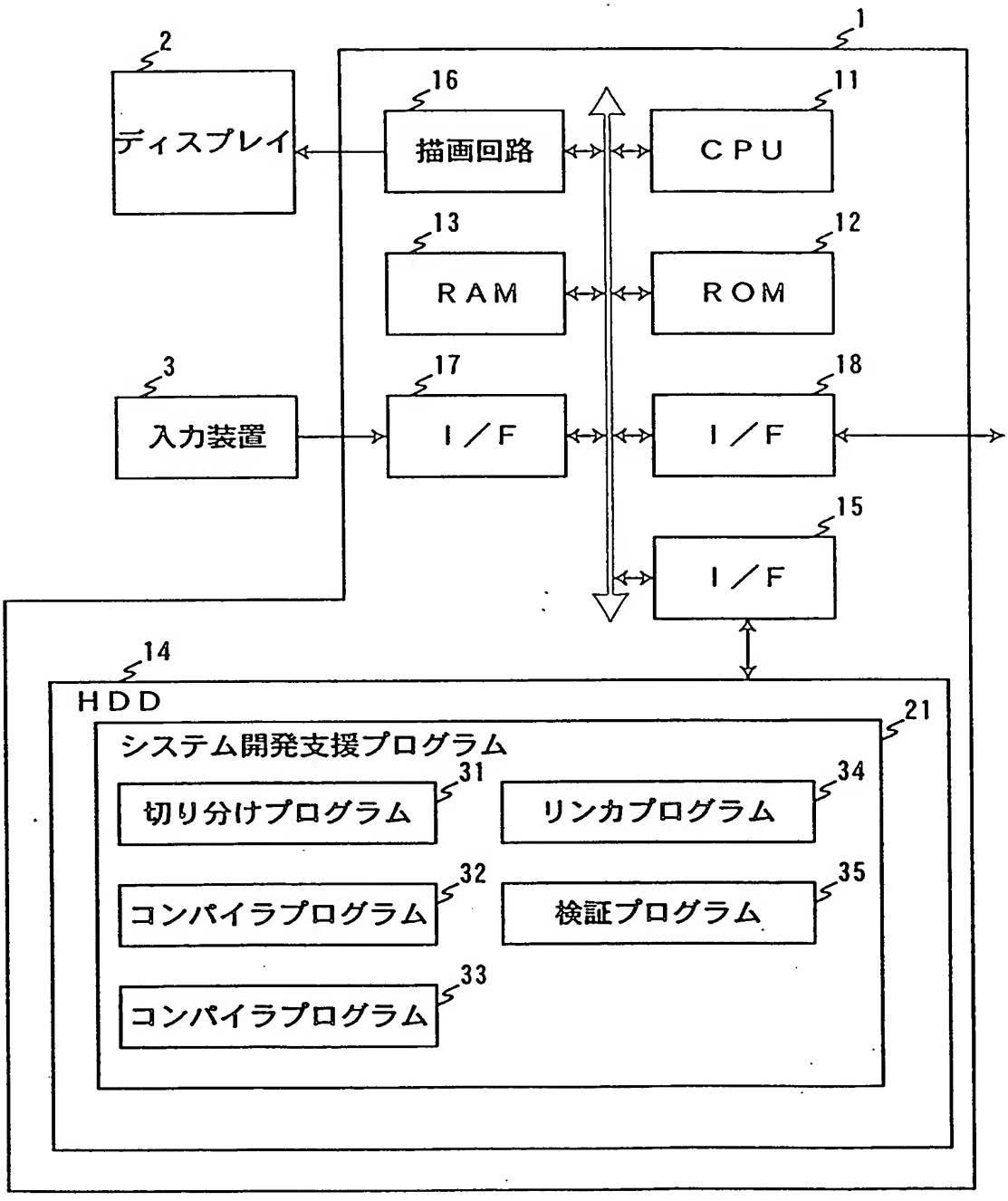
としてコンピュータを機能させるためのシステム開発支援プログラムを記録したコンピュータ読み取り可能な記録媒体。

１５．プログラムの各部をハードウェア部分およびソフトウェア部分のいずれかに指定する切り分け情報に基づいて、システムの論理仕様が単一の高級言語で記述されているプログラムをハードウェア部分とソフトウェア部分とに切り分け、そのハードウェア部分のプログラムおよびそのソフトウェア部分のプログラムを記憶手段に記憶させる切り分け手段としてコンピュータを機能させるための切り分けプログラムを記録したコンピュータ読み取り可能な記録媒体。

要　約　書

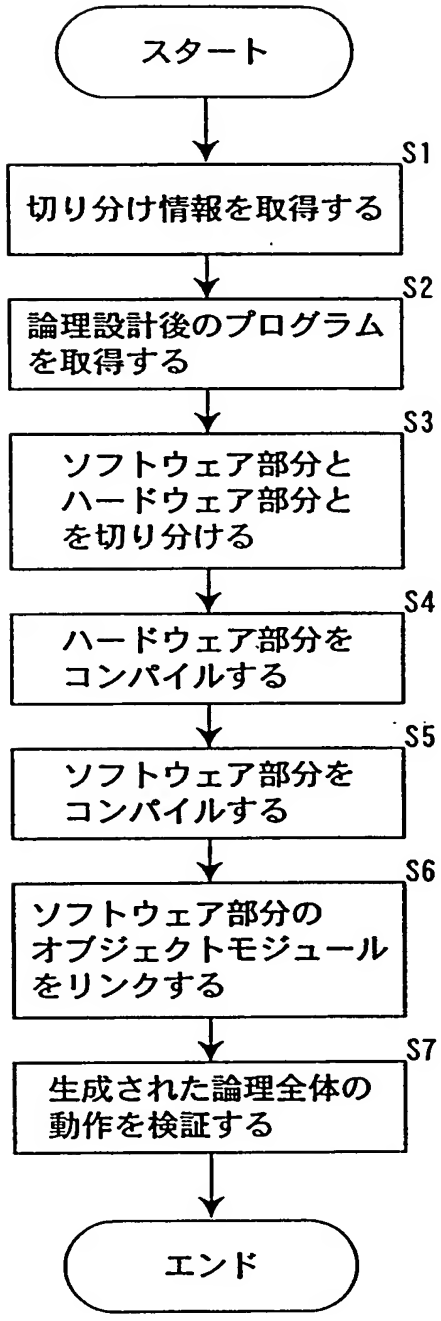
切り分け手段は、プログラムの各部をハードウェア部分およびソフトウェア部分のいずれかに指定する切り分け情報に基づいて、システムの論理仕様が単一の高級言語で記述されているプログラムをハードウェア部分とソフトウェア部分とに切り分ける。記憶手段は、切り分け手段により切り分けられたハードウェア部分のプログラムおよびソフトウェア部分のプログラムを記憶する。第１の変換手段は、記憶手段に記憶されたハードウェア部分のプログラムを回路仕様に変換する。第２の変換手段は、記憶手段に記憶されたソフトウェア部分のプログラムを実行形式モジュールに変換する。

第１図


1
2
ディスプレイ
16
描画回路
11
CPU
13
RAM
12
ROM
3
入力装置
17
I／F
18
I／F
15
I／F
14
HDD
21
システム開発支援プログラム
31
切り分けプログラム
34
リンカプログラム
32
コンパイラプログラム
35
検証プログラム
33
コンパイラプログラム

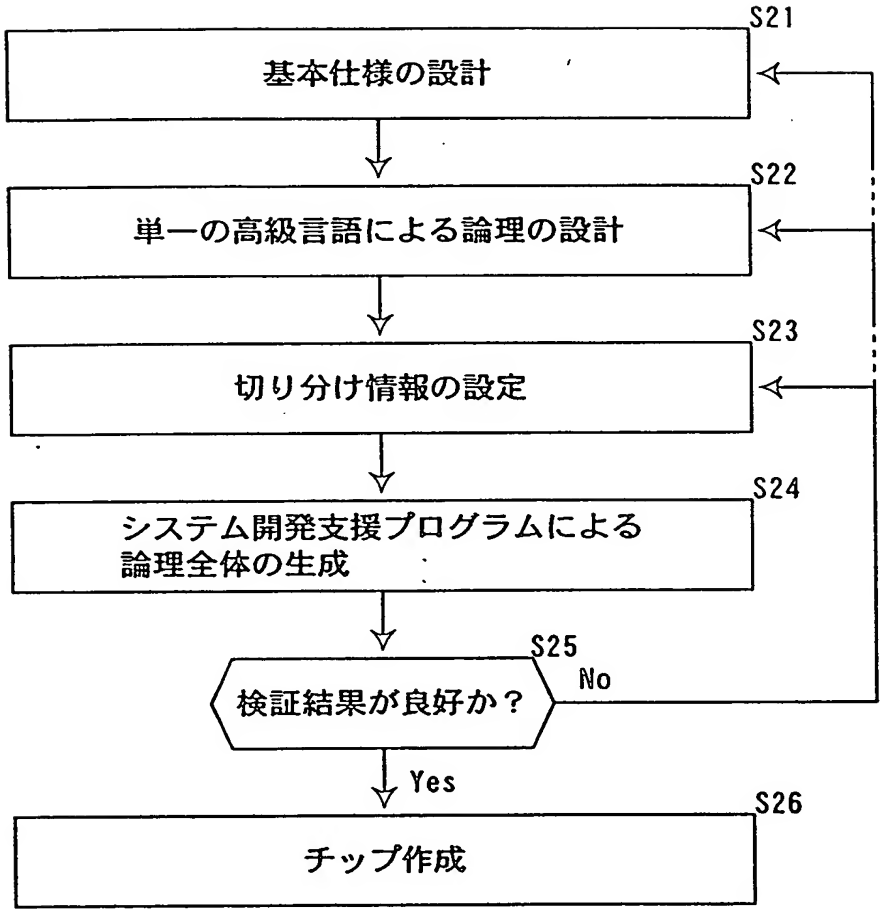
THIS PAGE BLANK (USPTO)

第２図


スタート
S1
切り分け情報を取得する
S2
論理設計後のプログラムを取得する
S3
ソフトウェア部分とハードウェア部分とを切り分ける
S4
ハードウェア部分をコンパイルする
S5
ソフトウェア部分をコンパイルする
S6
ソフトウェア部分のオブジェクトモジュールをリンクする
S7
生成された論理全体の動作を検証する
エンド

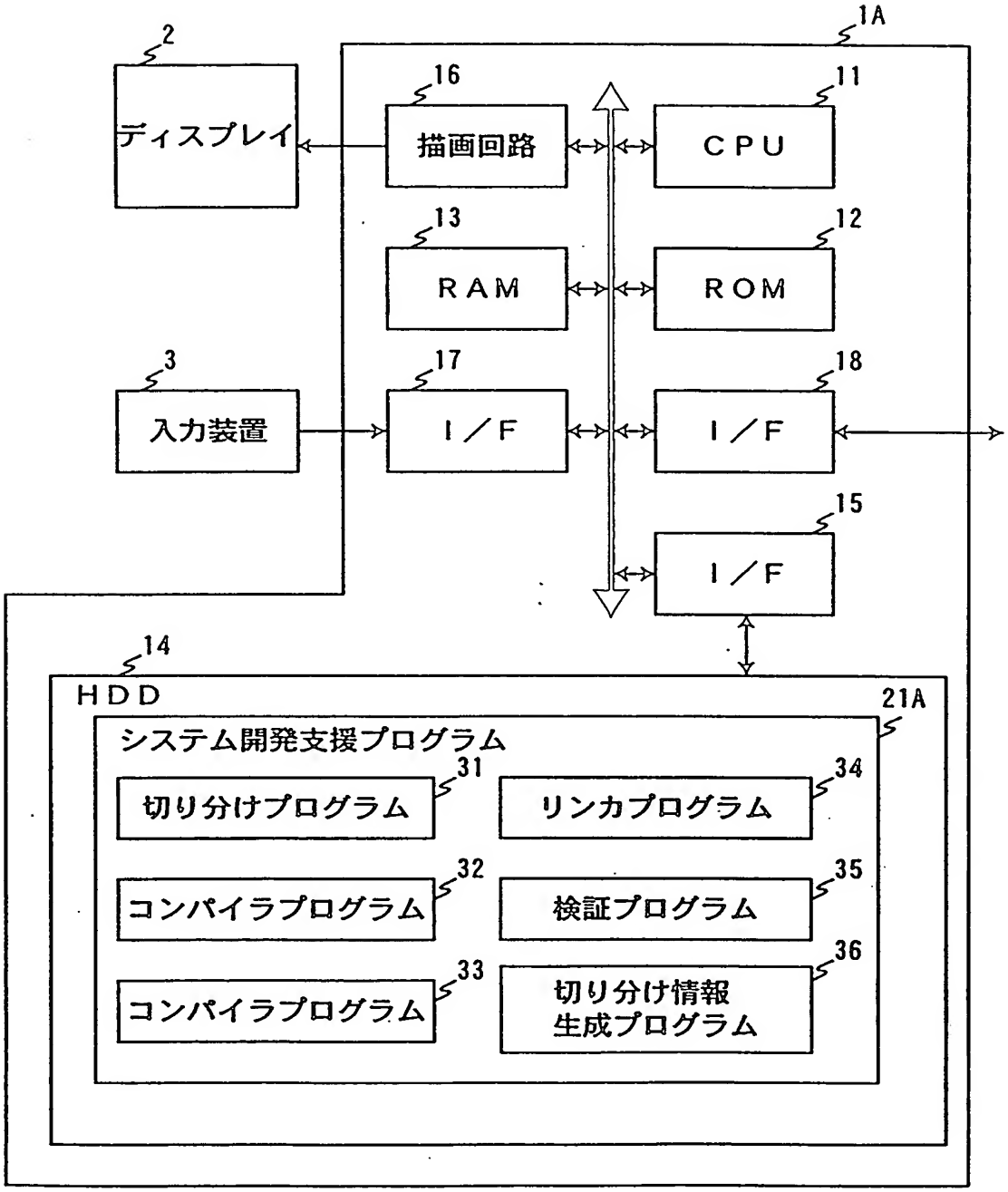
THIS PAGE BLANK (USPTO)

第３図


S21
基本仕様の設計
S22
単一の高級言語による論理の設計
S23
切り分け情報の設定
S24
システム開発支援プログラムによる
論理全体の生成
S25
検証結果が良好か？
No
Yes
S26
チップ作成

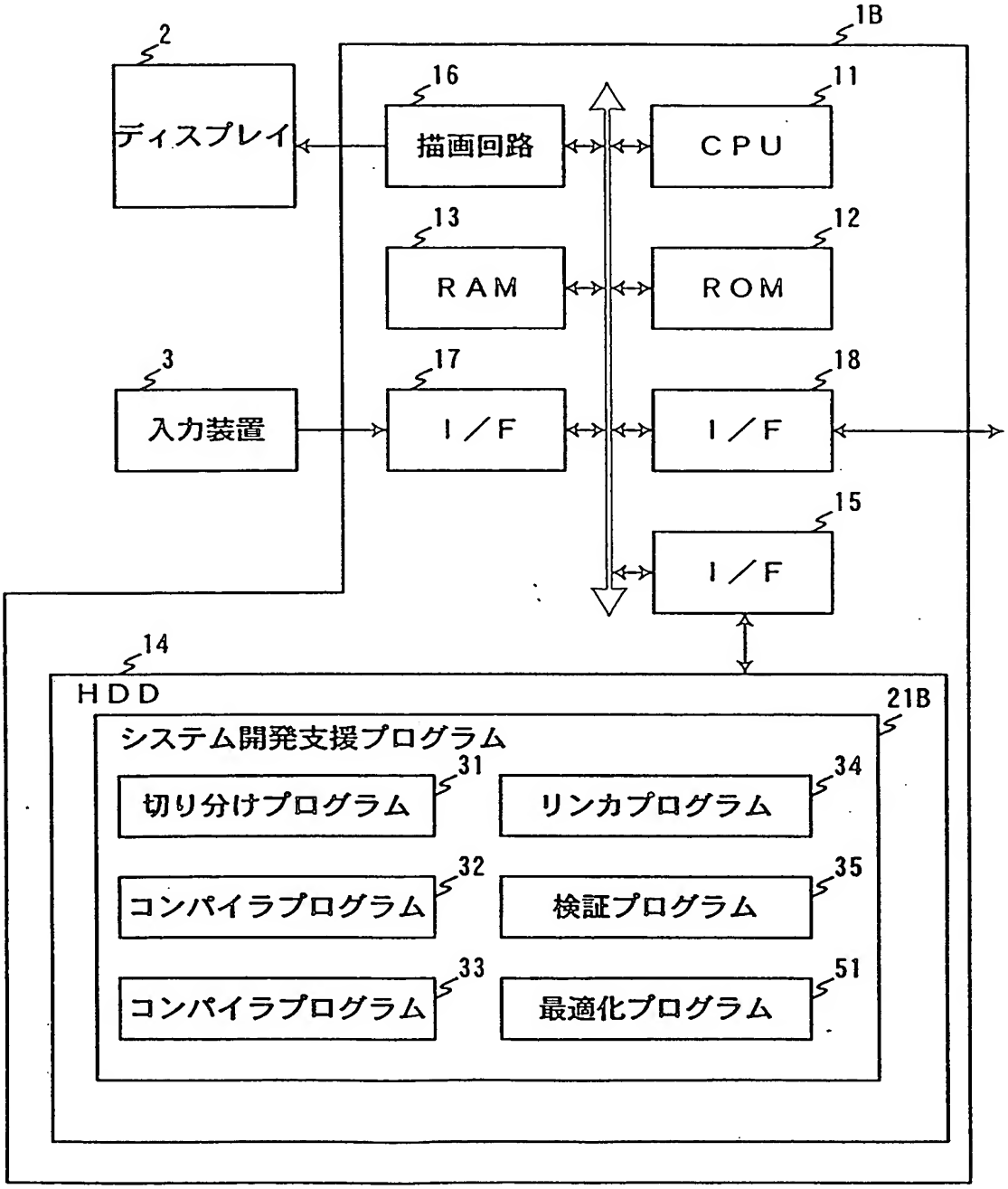
THIS PAGE BLANK (USPTO)

第４図


1A
2
ディスプレイ
16
描画回路
11
CPU
13
RAM
12
ROM
3
入力装置
17
I／F
18
I／F
15
I／F
14
HDD
21A
システム開発支援プログラム
31
切り分けプログラム
34
リンカプログラム
32
コンパイラプログラム
35
検証プログラム
33
コンパイラプログラム
36
切り分け情報
生成プログラム

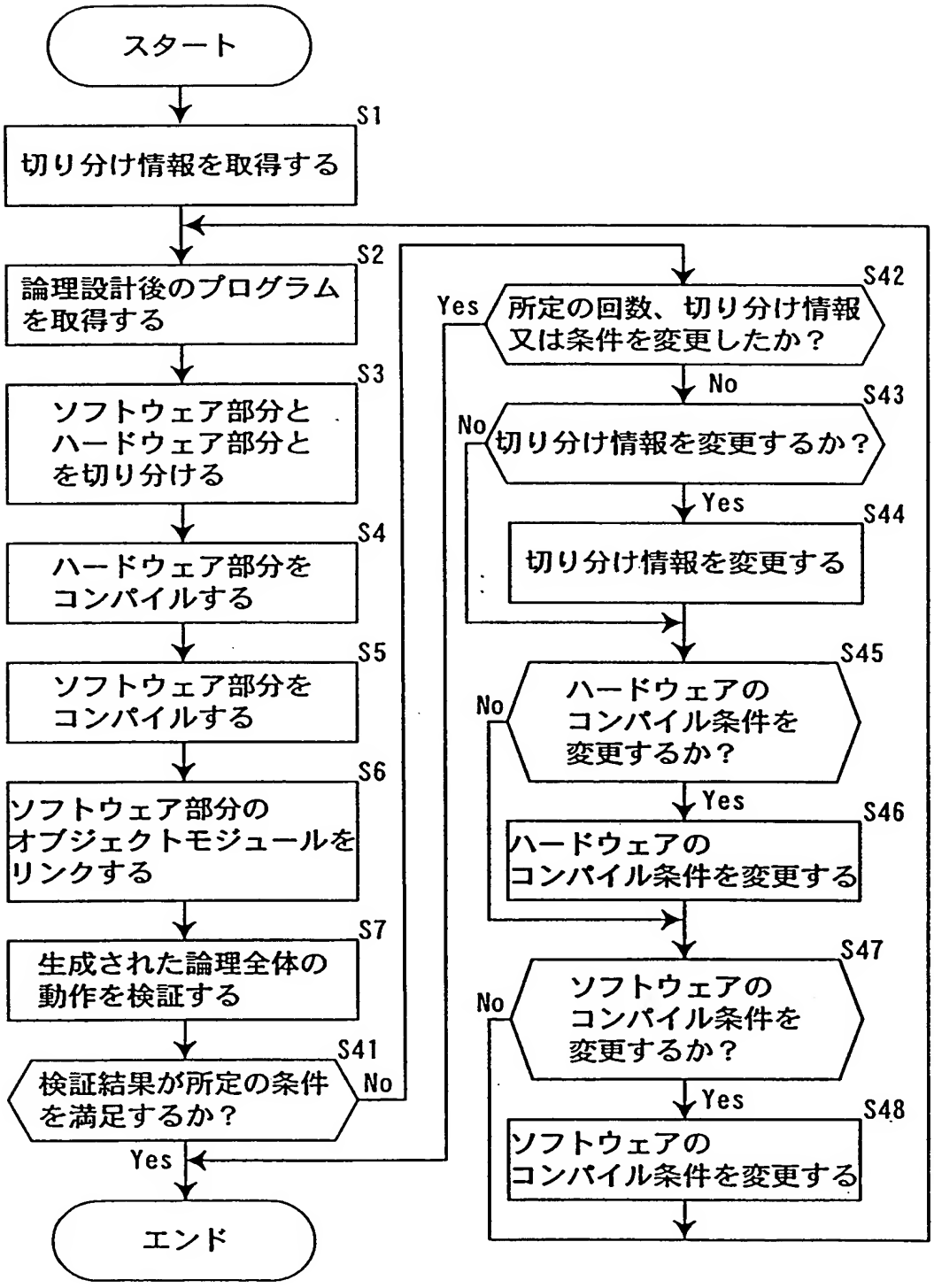
THIS PAGE BLANK (USPTO)

第５図


1B
2
ディスプレイ
16
描画回路
11
CPU
13
RAM
12
ROM
3
入力装置
17
I／F
18
I／F
15
I／F
14
HDD
21B
システム開発支援プログラム
31
切り分けプログラム
34
リンカプログラム
32
コンパイラプログラム
35
検証プログラム
33
コンパイラプログラム
51
最適化プログラム

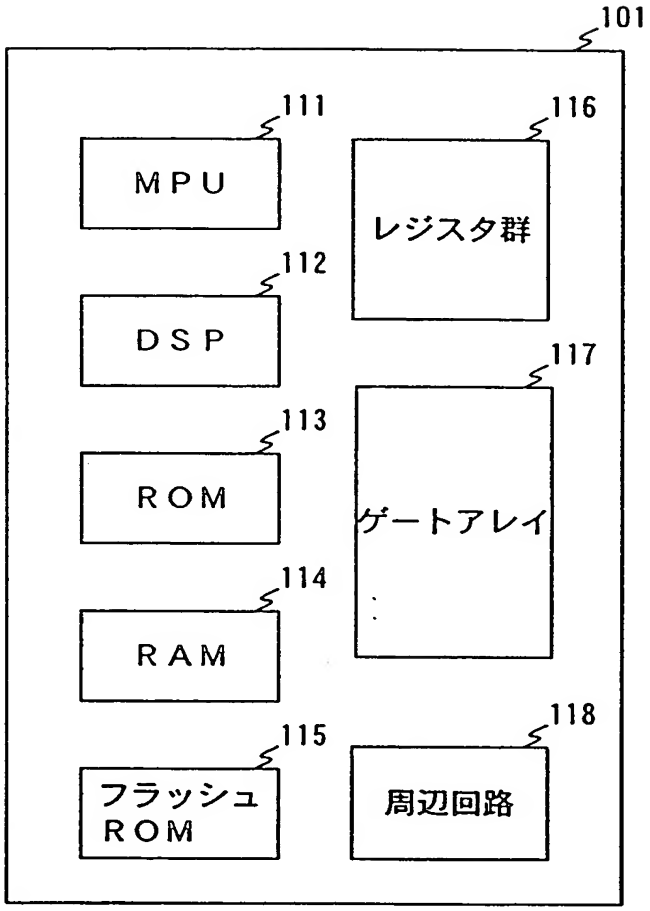
THIS PAGE BLANK (USPTO)

第6図


スタート
S1
切り分け情報を取得する
S2
論理設計後のプログラムを取得する
S3
ソフトウェア部分とハードウェア部分とを切り分ける
S4
ハードウェア部分をコンパイルする
S5
ソフトウェア部分をコンパイルする
S6
ソフトウェア部分のオブジェクトモジュールをリンクする
S7
生成された論理全体の動作を検証する
S41
検証結果が所定の条件を満足するか？
No
Yes
エンド
S42
Yes
所定の回数、切り分け情報又は条件を変更したか？
No
S43
No
切り分け情報を変更するか？
Yes
S44
切り分け情報を変更する
S45
No
ハードウェアのコンパイル条件を変更するか？
Yes
S46
ハードウェアのコンパイル条件を変更する
S47
No
ソフトウェアのコンパイル条件を変更するか？
Yes
S48
ソフトウェアのコンパイル条件を変更する

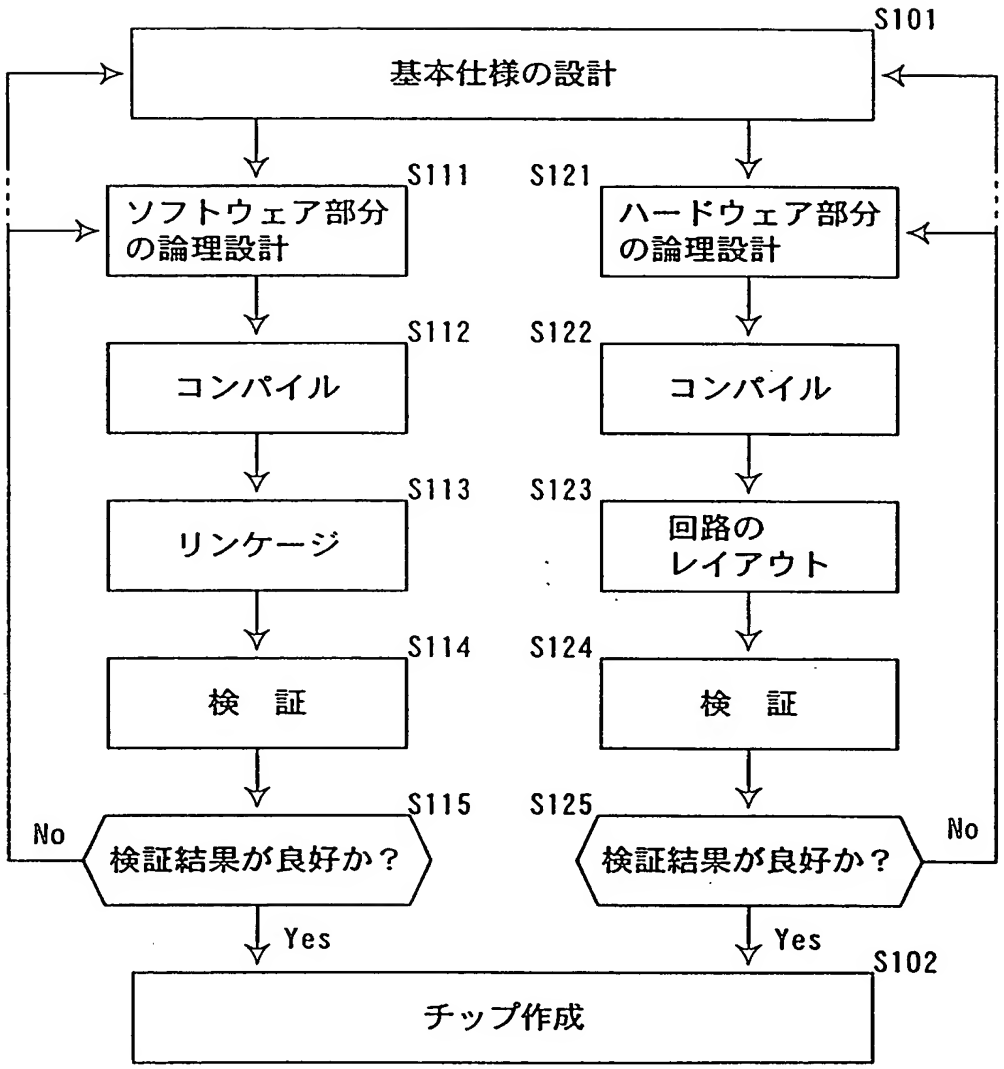
THIS PAGE BLANK (USPTO)

第７図


101
111
MPU
112
DSP
113
ROM
114
RAM
115
フラッシュ
ROM
116
レジスタ群
117
ゲートアレイ
118
周辺回路

THIS PAGE BLANK (USPTO)

第８図


S101
基本仕様の設計
S111
ソフトウェア部分の論理設計
S121
ハードウェア部分の論理設計
S112
コンパイル
S122
コンパイル
S113
リンケージ
S123
回路のレイアウト
S114
検　証
S124
検　証
S115
S125
No
検証結果が良好か？
No
検証結果が良好か？
Yes
Yes
S102
チップ作成

THIS PAGE BLANK (USPTO)